

A

Please type a plus sign (+) inside this box [+]

PTO/SB/05 (12/97)

Approved for use through 09/30/00. OMB 0651-0032

Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Attorney Docket No. 02950.P033 Total Pages 3First Named Inventor or Application Identifier Paul E. MatzExpress Mail Label No. EL370843766USADDRESS TO: Assistant Commissioner for Patents
Box Patent Application
Washington, D. C. 20231**APPLICATION ELEMENTS**

See MPEP chapter 600 concerning utility patent application contents.

1. X Fee Transmittal Form
(Submit an original, and a duplicate for fee processing)
2. X Specification (Total Pages 32)
(preferred arrangement set forth below)
 - Descriptive Title of the Invention
 - Cross References to Related Applications
 - Statement Regarding Fed sponsored R & D
 - Reference to Microfiche Appendix
 - Background of the Invention
 - Brief Summary of the Invention
 - Brief Description of the Drawings (if filed)
 - Detailed Description
 - Claims
 - Abstract of the Disclosure
3. X Drawings(s) (35 USC 113) (Total Sheets 10)
4. Oath or Declaration (Total Pages)
 - a. Newly Executed (Original or Copy)
 - b. Copy from a Prior Application (37 CFR 1.63(d))
(for Continuation/Divisional with Box 17 completed) (**Note Box 5 below**)
 - i. **DELETIONS OF INVENTOR(S)** Signed statement attached deleting inventor(s) named in the prior application, see 37 CFR 1.63(d)(2) and 1.33(b).
5. Incorporation By Reference (useable if Box 4b is checked)
The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied under Box 4b, is considered as being part of the disclosure of the accompanying application and is hereby incorporated by reference therein.
6. Microfiche Computer Program (Appendix)

12/01/97

- 1 -

PTO/SB/05 (12/97)

05/26/99

Jc660 U.S. PTO

Jc549 U.S. PTO

09/30/99

05/26/99

7. _____ Nucleotide and/or Amino Acid Sequence Submission
(if applicable, all necessary)
a. _____ Computer Readable Copy
b. _____ Paper Copy (identical to computer copy)
c. _____ Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

8. _____ Assignment Papers (cover sheet & documents(s))
9. _____ a. 37 CFR 3.73(b) Statement (where there is an assignee)
_____ b. Power of Attorney
10. _____ English Translation Document (if applicable)
11. _____ a. Information Disclosure Statement (IDS)/PTO-1449
_____ b. Copies of IDS Citations
12. _____ Preliminary Amendment
13. X Return Receipt Postcard (MPEP 503) (Should be specifically itemized)
14. _____ a. Small Entity Statement(s)
_____ b. Statement filed in prior application, Status still proper and desired
15. _____ Certified Copy of Priority Document(s) (if foreign priority is claimed)
16. X Other: Declaration/Power of Attorney
Associate Power of Attorney
Letter Regarding Limited Recognition

17. **If a CONTINUING APPLICATION**, check appropriate box and supply the requisite information:

____ Continuation ____ Divisional ____ Continuation-in-part (CIP)
of prior application No: ____

18. **Correspondence Address**

____ Customer Number or Bar Code Label
____ (Insert Customer No. or Attach Bar Code Label here)
or

X Correspondence Address Below

NAME BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

ADDRESS 12400 Wilshire Boulevard
Seventh Floor

CITY Los Angeles STATE California ZIP CODE 90025-1026

Country U.S.A. TELEPHONE (408) 720-8598 FAX (408) 720-9397

12/01/97

- 2 -

PTO/SB/05 (12/97)

Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

UNITED STATES PATENT APPLICATION

FOR

**METHODS AND APPARATUS FOR EXECUTING A TRANSACTION
TASK WITHIN A TRANSACTION PROCESSING SYSTEM
EMPLOYING SYMMETRIC MULTIPROCESSORS**

INVENTORS:

Paul E. Matz

Glen K. Okita

Gebran Chahrouri

Michael Butensky

Prepared by:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CALIFORNIA 90025
(408) 720-8598

Attorney's Docket No. 02950.P033

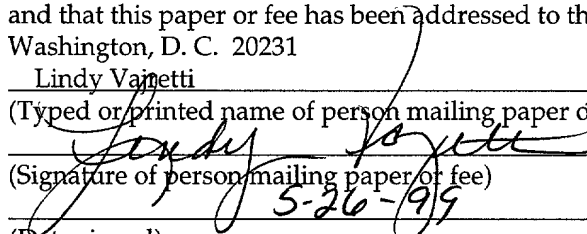
"Express Mail" mailing label number: EL370843766US

Date of Deposit: May 26, 1999

I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to the Assistant Commissioner for Patents, Washington, D. C. 20231

Lindy Vajetti

(Typed or printed name of person mailing paper or fee)



(Signature of person mailing paper or fee)

5-26-99

(Date signed)

METHODS AND APPARATUS FOR EXECUTING A TRANSACTION TASK WITHIN A TRANSACTION PROCESSING SYSTEM EMPLOYING SYMMETRIC MULTIPROCESSORS

5

FIELD OF THE INVENTION

The present invention relates generally to the field of transaction processing. More specifically, the present invention relates to the executing of a transaction task within a transaction processing system employing a multiprocessor (MP) architecture.

10

BACKGROUND OF THE INVENTION

Historically, transaction processing systems, such as for example Automatic Call Distributors (ACDs), have employed multiple processors and multiple operating systems for managing various tasks, including call routing, within such transaction processing systems. For example, in an exemplary ACD, a single processor and a single operating system may be dedicated to servicing non-critical tasks, such as historical and real-time reporting, database administration and system maintenance tasks. A further single processor and a single operating system within the ACD may then be dedicated to servicing real-time, critical tasks, such as ring no answer timing and other central office signaling tasks. Accordingly, the different operating systems may be utilized for servicing the respective non-critical tasks and the real-time, critical tasks. For example, the reporting, administration and maintenance tasks may be performed by a multipurpose operating system

15

20

25

such as Unix. On the other hand, the real-time, critical tasks may be performed by a real-time operating system such as the VxWorks operating system developed by Wind River Systems, Inc. of Alameda, California, the PSOS operating system or the Lynx operating system. By restricting the

5 execution of tasks to a particular processor and a particular operating system, a transaction processing system may be unable to respond to peak performance demands in certain situations.

SUMMARY OF THE INVENTION

According to the present invention, there is provided a method of executing a transaction task within a transaction processing system.

Responsive to an event, a workflow associated with the event is identified.

- 5 A transaction task, that at least partially executes the workflow, is distributed to an available thread within a pool threads operating within a multiprocessor system.

Other features of the present invention will be apparent from the accompanying drawings and from the detailed description which follows.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limitation in the figures of the accompanying drawings, in which like references indicate similar elements and in which:

5

Figure 1 is a block diagram illustrating a transaction processing system wherein separate and distinct transaction processing subsystems are dedicated to handling different types of tasks.

10

Figure 2 is a block diagram illustrating a transaction processing system, according to an exemplary embodiment of the present invention, within which the present invention may be implemented and performed.

15

Figure 3 is a block diagram illustrating an exemplary transaction processing system in the form of a multi-site call center environment that may include a number of the transaction processing systems shown in **Figure 2**.

20

Figure 4 is a block diagram illustrating a call center site, according to an exemplary embodiment of the present invention, having an alternative architecture from the call center site shown in **Figure 3**.

Figures 5A and 5B are block diagrams illustrating a workflow

execution system, according to an exemplary embodiment of the present invention, that may be employed within a workflow server engine or workflow router described with reference to **Figure 4**.

5 **Figure 6** is a block diagram illustrating the structure of an exemplary event object.

Figure 7 is a block diagram illustrating the structure of an exemplary task object.

10

Figure 8 is a flowchart illustrating a method, according to an exemplary embodiment of the present invention, of creating task object that is queued within a task queue illustrated in **Figure 5**.

15

Figure 9 is a flowchart illustrating a method, according to an exemplary embodiment of the present invention, of executing a transaction task within a multiprocessor system, such as a Symmetrical Multiprocessor (SMP) system.

20

Figure 10 is a block diagram providing a conceptual representation of the creation of event and task objects.

Other features of the present invention will be apparent from the

accompanying drawings and from the detailed description which follows.

02950.P033

DETAILED DESCRIPTION

A method and apparatus for executing a transaction task within a multiprocessor (MP) transaction processing system are described. In the following description, for the purposes of explanation, numerous specific
5 details are set forth in order to provide a thorough understanding of the present invention. It will be evident, however, to one skilled in the art that the present invention may be practiced without these specific details.

For the purposes of the present specification, the term "workflow" shall be taken to mean a sequence of steps that are performed to, at least
10 partially, process a transaction. Further, the term "task" shall be taken to mean a process, method, operation or step that implements the performance of a workflow sequence. A task may furthermore execute a series of "sub-tasks".

The term "thread" shall be taken to refer to any entity to which an
15 operating system allocates processor time within a computer system. Optionally, a thread may execute any part of an application's code, including a part currently being executed by another thread instance. All threads of a process may share a virtual address space, global variables, and operating system resources of the process. A process may include one or
20 more threads that run in the context of the process. A "process" may be an application that may include a private virtual address space, code, data and other operating system resources (e.g., files, pipes and synchronization objects that are visible to the process).

Exemplary Transaction Processing Systems

Figure 1 is a block diagram illustrating a transaction processing system 10 wherein separate and distinct transaction processing subsystems 12 and 14 are dedicated to handling different types of tasks. Specifically, in the subsystem 12, a single central processing unit (CPU) 16 and a single operating system 18 service a number of non-critical applications, such as for example, a historical reporting application, a database administration application, and a system maintenance application. In the subsystem 14, a single CPU 20 and a single operating system 22 service a number of real-time, critical applications, such as a central office signaling application. A transaction processing system 10, such as that illustrated in **Figure 1**, may prove to have inadequate resources to respond to peak performance demands. For example, the subsystem 14 may prove incapable of handling an unusually high level of central office signaling.

Figure 2 is a block diagram illustrating a transaction processing system 30, such as for example an ACD, including a multiprocessor (MP) transaction processing subsystem 31 within which a method, according to an exemplary embodiment of the present invention, of executing a transaction task may be performed. The transaction processing subsystem 31 may be dedicated to performing a specific task within the transaction processing system 30, such as for example transaction routing. Examples of transaction routing include telephone call routing, e-mail routing, and Web request

routing, as are described in further detail below, from a source to a software or human agent.

The transaction processing subsystem 31 may employ a Symmetric Multiprocessing (SMP) architecture and include a bank of processors 32 that
5 share a memory 34 and an input/output (I/O) subsystem 36. The bank of processors 32 may include between two (2) and thirty-two (32) processors, each of which may be an Intel Pentium® Pro or Pentium II® processor manufactured by Intel Corp. of Santa Clara, California, or a SPARC microprocessor manufactured by Sun Microelectronics of Mountain View,
10 California. The processors 32, the shared I/O subsystem 36 and the shared memory 34 are all controlled by a single executing SMP-enabled operating system 38 that resides in the shared memory 34. Examples of an SMP-enabled operating system 38 include the Windows NT® operating system developed by Microsoft Corp. of Redmond, Washington state, the OS/2
15 operating system developed by IBM Corp., or a variant of the Unix operating system, such as the Solaris® operating system. The shared memory 34 is furthermore shown to host both non-critical and critical real-time applications, such as a reporting application 40, an administrative application 42 and a signaling application 44.

20 In a further embodiment of the present invention, the transaction processing system 30 may comprise a clustered SMP system, in which case a number of SMP systems, such as that illustrated as 31, may be included within the transaction processing system 30.

Exemplary Transaction Processing Environment

Figure 3 is a block diagram illustrating an exemplary transaction processing environment 50, in the form of a multi-site call center environment, that may include multiple transaction processing systems 30, such as that shown in **Figure 2**. Specifically, **Figure 3** provides an enterprise-wide view of the transaction processing environment 50 that includes two call center sites 52 and 54 that may be coupled via a Wide Area Network (WAN) 56 to each other and to customer enterprise systems 58, an enterprise workflow server 60 and an information server 62. The customer enterprise systems 58 may execute host-based Computer Telephony Integration (CTI) applications, such as "screen pops" and database lookup applications. The pre-routing workflow server 60 may perform a "pre-call routing" function. Merely for example, the workflow server 60 may collect information from multiple call center sites in a multi-site heterogeneous or homogeneous call center environment, interpret this information, and provide routing information to a Service Control Point (SCP) regarding where to route a call based on the call center site data and user preferences. Accordingly, the workflow server 60 provides a framework for a single system image view of resources on multiple call center sites, and the capability to route a call to a specific call center site based on resources, skills and agent availability at the multiple call center sites. Such routing decisions may be based on near real-time data collected from the respective call center sites, and on the routing preferences specified by users. The information server 62 also provides real-

time and enterprise-wide information to a multi-site call center system administrator, and may also gather information for the purposes of near real-time management of a multi-site call center environment shown in **Figure 3.**

5 Each of the call center sites 52 and 54 is equipped to receive transaction requests (e.g., calls, e-mails or network requests) over a variety of media, and to process and facilitate transactions between, for example, a source and a human (or software) agent responsive to such transaction requests. To this end, each of the call center sites 52 and 54 is shown to
10 include a number of transaction processing systems, namely an e-mail server 64, a web server 66, an Interactive Voice Response (IVR) workflow server 68, an ACD 70, a Computer Telephony Integration (CTI) server 72 and a workflow server 74. Each call center site 52 and 54 is also shown to include a telephony device server 76, an agent access server 78 and agent desktop
15 clients 80. The ACD 70 is also shown to include call processing functionality 83, whereby telephone calls (e.g., both switched and voice-over-IP calls) may be processed and routed to an available agent teleset (not shown).

Each of the call center sites 52 and 54 also includes a number of administrative clients 82, whereby a call center site administrator may
20 configure and edit workflow definitions that define workflows that are executed by various workflow servers within the respective call center sites.

Figure 4 is a block diagram illustrating a call center site 90, according to an exemplary embodiment of the present invention, having an alternative

architecture to the call center sites 52 and 54 illustrated in **Figure 3**.

Specifically, the workflow server engines for directing and routing calls within the call center sites 52 and 54 of **Figure 3** are shown to be distributed over a number of transaction processing systems, such as the workflow
5 server 74, the IVR workflow server 68, a call center customer relationship server 71, and the CTI server 72. The call center site 90 illustrated in **Figure 4** provides a more integrated environment in which a single workflow server engine 92 within the server 71 routes transaction information over a wide variety of media. The workflow server engine 92 is a provided with "events"
10 by a number of event subsystems 94 that receive input from a number of servers, such as for example in the e-mail server 64, the web server 66, and a video server 69. The event subsystem 94 also provides events to the workflow server engine 92 resulting from telephone calls received at a telephony device server 76 via the Public Switched Telephone Network
15 (PSTN) 77 or via the Internet 79.

The call center site 90 includes a number of agent stations 98, each of which may include a teleset 100 via which a human agent may respond to transaction requests received via any of the media servers and a collection of agent desktop applications 102 for facilitating transaction processing over,
20 for example, the Internet utilizing e-mail or the World Wide Web (WWW). For example, the agent desktop applications 102 may include an e-mail client, a browser client, a web collaboration client and a video conferencing client. These agent desktop applications may be highly integrated, or may

be stand-alone applications. Alternatively, each agent station 98 may comprise a software agent, which is able to respond to transaction requests, and process a transaction to completion, in an automated fashion.

The present invention will be described below within the context of a workflow router, which includes a workflow server engine. It will be appreciated that the teachings of the present invention may be applied to any one of the workflow servers, workflow server engines, or call processing functions illustrated in **Figure 3**. Further, the teachings of the present invention are also applicable to the "pre-call routing" workflow servers and "post-call routing" workflow servers that may be employed within a transaction processing environment.

Exemplary Workflow Execution System

Figures 5A and 5B are block diagrams illustrating a workflow execution system 120, according to an exemplary embodiment of the present invention, that may be employed within any one of the workflow server engines or workflow routers described above. The workflow execution system 120 includes a workflow execution server 122 and a database server 124. The execution server 122 includes a number of event subsystems 126 (also termed "event providers") that generate tasks for a task queue 128 responsive to external transaction occurrences that are communicated to the event subsystem 126 as messages from appropriate clients. Such tasks may be any tasks required for the facilitating of a transaction and for fulfilling system requirements within a transaction processing system. While such

tasks are described below in the context of routing tasks (for routing a transaction to an agent), the tasks could include data storage and retrieval tasks that store and retrieve data pertinent to the transaction. The tasks may also include tasks that facilitate interaction with agents, such as “screen pop” generation. Tasks may also include reporting, maintenance or system administration tasks. Transaction occurrences may include, for example, the receipt of a transaction request (e.g., an e-mail or telephone call), the termination of a transaction by a source (e.g., a client hangs up prior to a queued telephone call being serviced), or a system failure or shutdown for some other reason. As shown in **Figure 5B**, each event subsystem 126 calls a re-entrant task dispatcher 200 that is responsible for creating a task object, or set of task objects that may be executed. The tasks are created responsive to reception of an event generated by the relevant subsystem. Specifically, if an event invokes a workflow, a task dispatcher 200 creates a task object that dispatches to, and queued within, the task queue 128 for later execution. To generate such task objects, a called task dispatcher 200 accesses workflow definitions 208, event definitions 210 and event-workflow binding information 214 stored within the database server 124. A pool of worker threads 202 executes tasks stored within the task queue 128. Task priority logic 230 may determine the priority of a task within the task queue 128 utilizing workflow priority information 216 and/or event priority information 217, both of which are stored within the database server 124. A database server interface 220 facilitates access by task dispatchers 200, the

task queue 128 and the task priority logic 230 to information stored in the database server 124. Task execution by the pool of worker threads 202 furthermore generates messages to a reporting service 222.

Event Subsystems

5 Each of the event subsystems 126 generates events by calling an event generator routine provided by the execution server 122. Each of the event subsystems 126 furthermore includes a unique subsystem identifier. In one embodiment, the event subsystems may furthermore be classified as being either (1) administrative event subsystems or (2) schedule event subsystems
10 providing respective administrative and schedule tasks to the tasks queue 128.

 An exemplary event 146, that may be generated by any one of the event subsystems, is illustrated in **Figure 6**, and is shown to include an event identifier 142, a subsystem identifier 144 identifying the subsystem that
15 generated the event 146, and a property list 140. The property list 140 is a set of variables or parameters represented as name-value pairs. The database server 124 includes a list of all event definitions 210.

 Exemplary event subsystems 126 include an administrative event subsystem that collects TCP/IP messages that control the execution server
20 122. These messages typically originate from administrative clients 82, such as a workflow builder 132 or an administration console 134, and include a command to be executed by the execution server 122 on the order of the relevant clients. Such commands may include commands directing the

execution server 122 (1) to start, stop, suspend, resume or step a workflow, (2) to modify a task being executed within the execution server 122, (3) to modify the number of worker threads included within a pool of such threads, (4) to add or remove an event-workflow binding, or (5) to suspend, resume or shutdown the execution server 122.

An exemplary telephony event subsystem 150 collects messages from, for example, a CTI server 72 regarding telephone calls received at that server. An exemplary schedule event subsystem 152 propagates tasks to the task queue 128 according to a schedule specified by, for example, the administrative console 134. The events generated by the schedule event subsystem 152 may be for any subsystem identifier and event identifier, and may also comprise command events. An exemplary pre-call routing subsystem 154 services pre-routing queries from the PSTN 77, and interacts with pre-routing clients using TCP/IP connection-oriented sockets to provide an interface between such pre-routing clients and the pre-call routing subsystem 154. Other event subsystems may include a web event subsystem 156 and an e-mail event subsystem 158.

Task Dispatcher and Task Queue

The workflow execution server 122 includes a single task queue 128 to manage tasks received from the task dispatchers 200. As noted above, each of the event subsystems 126 generates events that are translated into tasks dispatched to the task queue 128. The tasks are prioritized within the task queue 128 by task priority logic 230, each task being assigned a default

priority of zero (0). The task queue 128 utilizes Adaptive Communication Environment (ACE) synchronization methods to ensure that multiple event subsystems 126 may properly share the task queue 128. ACE is a freely available C++ framework, and provides abstractions for sockets, queues and
5 high-level components. ACE is distributed by Douglas Schmidt at Washington University, and further details regarding ACE can be found at: <http://www.cs.wust.edu/~schmidt/ACE.html>.

Each task dispatcher 200 furthermore uses ACE notification methods to effectively dispatch tasks to the task queue 128. Specifically, a task
10 dispatcher 200 may look to an event header to determine how to handle the relevant event. If the event is identified as being a workflow event, the task dispatcher 200 matches the event to an associated workflow utilizing the event-workflow binding information 214 stored in the database server 124. The task dispatcher 200 utilizes the subsystem identifier 144 and the event
15 identifier 142 of a relevant event to identify an associated workflow. More than one event-workflow binding may be located. If a matching workflow (or set of workflows) is identified, the workflow (or set of workflows) is instantiated by the task dispatcher 200 to create a task object (or multiple task objects) to execute the workflow(s). These task objects are dispatched to
20 the task queue 128. It should thus be noted that an event may have multiple tasks associated therewith.

In addition to workflow centers that are mapped to workflows using the event-workflow binding information 214 in the manner described above,

663020200
further event types exist that may conveniently be classified as "task" events
and events that may be classified as "command" events. A valid task
identifier (not shown) distinguishes a task event 146 in an event header that
identifies an associated task. The task dispatcher 200 dispatches a task event
5 to a task specified and identified by the task identifier. Task events send
events to an executing task and do not invoke, create or start new tasks. A
command event 146 is dispatched by the task dispatcher 200 to a command
interpreter (not shown) to execute an included command. A command
event may be handled by the pool of worker threads 202, or may
10 alternatively be for a subsystem.

Task Priority Logic

Within the task queue 128, each task 250 has a unique task identifier
252 associated therewith. **Figure 7** is a block diagram illustrating an
exemplary task 250, and shows the task identifier 252. Each task 250 may
15 furthermore be assigned a priority 254 corresponding to the priority of the
event that generated the task. In one embodiment, workflows (as defined by
the workflow definitions 208 in the database server 124) may each have
priorities associated therewith, and as recorded in the workflow priority
information 216, that override the priorities associated with a task based on
20 an event priority. Again, if no priority for a task is specified, a default
priority of zero (0) may be assigned to a task 250. Each task 250 is
furthermore shown to include a reference (e.g., a pointer) 256 to a current
step or a number of steps that constitute the task 250, a variable context 258

(i.e., data), a pointer to the workflow from which the task 250 was instantiated, at least one method 262 used to execute the task, and a processor affinity 264 identifying a processor within a multiprocessor environment on which the task 250 should preferably be executed. When a sub-task is executed, a stack of the original task and subsequent sub-tasks is maintained for each task object.

Pool of Worker threads

The pool of worker threads 202 is responsible for executing the tasks 205 queued within the task queue 128. As each worker thread becomes available, a scheduler 204 identifies the highest priority task from the task queue 128, and feeds the task to the available worker thread that executes a single step of the relevant task. Further details regarding the execution of tasks by the pool of threads, where the pool of threads are executed on a multiprocessor platform, are provided below.

In an alternative embodiment of the present invention, an algorithm implemented within a scheduler associated with the task queue 128 may intelligently determined a "BestMatch" between an available thread and the tasks that are queued within the task queue 128. This "BestMatch" determination may be based on any number of parameters, such as a dynamically assigned priority or processor affinity.

In identifying a task to be attributed to an available worker thread, the scheduler 204 may identify a "real-time" priority associated with a task. Specifically, a task identified as having a "real-time" priority will be

regarded as having a highest priority, and assigned to an available thread ahead of any other tasks not having a "real-time" priority. In one embodiment of present invention, specific threads may be members of a "real-time" process priority class, and a task having a "real-time" priority will
5 be attributed to such threads by the scheduler.

Database Server Interface

As illustrated in **Figure 5**, the execution server 122 has a database connection via the database server interface 220 to the database server 124. In one embodiment of the present invention, this connection to the database
10 server 124 may be via a Remote Procedure Call (RPC) interface. Upon initialization of the workflow execution server 122, data is loaded from the database server 124. Specifically, the data required at initialization by the workflow execution server 122 includes (1) event-to-workflow bindings, (2) workflow definitions, (3) event definitions, (4) event schedules, and (5)
15 execution server parameters (e.g., thread pool size).

Methodology-Task Creation

Figure 8 is a flowchart illustrating a method 300, according to an exemplary embodiment of the present invention, of creating a task that is queued within the task queue 128. The method 300 commences at 302, and
20 proceeds to step 304, where an event occurrence is identified by an event subsystem 126. Merely for example, an ACD 70, shown in **Figure 3**, on receipt of a telephone call, may request a route over a CTI link to an agent.

At step 306, the relevant event subsystem (e.g., the telephony subsystem 150) generates an event, such as that illustrated in **Figure 6**. The event subsystem attributes a priority level to the event based on event content and/or event type. At step 310, the task dispatcher 200 called by the event subsystem
5 determines an event-workflow binding utilizing the event-workflow binding information 214 that was downloaded to the execution server 122 at initialization. At step 312, the task dispatcher 200 creates a task 250 (and dispatches the task to queue 128) by creating an instantiation of the workflow identified as being associated with the relevant event. The task
10 250 may be attributed a priority, as described above, based on the priority 314 of an underlying event or on a priority 316 assigned to the workflow. The method 300 then terminates at step 318.

Methodology-Task Execution

Figure 9 is a flowchart illustrating a method 350, according to an
15 exemplary embodiment of the present invention, of executing a task 250 within a multiprocessor system, such as for example a Symmetrical Multiprocessor (SMP) system. The method 350 commences at 352, and proceeds to step 354, where a worker thread within the pool of threads 202 becomes available as a result of completion of a preceding task. The
20 available worker thread then receives a task having a highest priority from the task queue 128, as identified by the scheduler 204. At step 356, the worker thread then executes one or more steps of the de-queued task. Specifically, a "dispatcher" within the kernel of an operating system, such as

the Windows NT ® operating system, assigns a thread to which the task is assigned to a processor within the multiprocessor system. In assigning a thread to a processor within such a multiprocessor system, the dispatcher will consider "thread affinity" that may specify a single processor, or group
5 of processors, on which the thread may execute. In an exemplary embodiment, thread affinity may be determined by a thread affinity mask, in the form of a bit vector representing the processors on which the respective thread is allowed to run. The thread affinity may also be specified by a process affinity mask, for a process within which the thread is
10 included, that comprises a bit vector specify processors on which the process is allowed to run.

Further, the dispatcher within the kernel of the operating system may recognize "real-time" priorities attributed to certain threads within the pool threads 202. Such threads may be dispatched to processors ahead of threads
15 having non-"real-time" priorities. This is especially applicable in a real-time operating system that guarantees interrupt latency or some other way for threads to obtain a guaranteed execution time.

At the same decision box 358, a determination is then made as to whether task is a "command" task for command execution. If so, the
20 relevant command is executed at step 360, whereafter the task is destroyed at step 362. Alternatively, should the task not be a command task, a determination is made at decision box 364 as to whether the task is a workflow task. If so, the next step of the relevant task is executed at step

366. Pending task notifications, indicated at 368, cause available exception handlers to set the next step. At decision box 370, a determination is made as to whether the step executed at step 366 was the last step of the task. If not, the method 350 proceeds to make a further determination at decision box 376 whether execution should continue for the same thread. If so, the method 350 loops back to step 366, and a next consecutive step of the relevant task is executed. Following a negative determination at decision box 374, the task is returned to, and again queued within, the task queue 128.

10 If the last step of the task has been executed, as recognized at decision box 370, the task is destroyed at step 362. The method 350 then terminates at step 372.

After all actions or steps associated with a task are completed, the thread then grabs the next available task from the task priority queue 128 for execution.

Accordingly, it will be appreciated that a task, which at least partially implements a workflow, is executed by any one of the worker threads within the pool of threads 202 that is available, or becomes available. Each of the worker threads within the pool 202 may execute on a designated processor within a bank of processors 32, such as that illustrated in **Figure 2**. In the absence of any processor or thread affinity associated with a task, the task may accordingly be executed on any one of the processors within the bank of processors 32. As events may be both non-critical or real-time

critical events, it will be appreciated that event types are not limited to being handled on one specific processor operating under the direction of one specific operating system. Accordingly, by allowing tasks generated responsive to events of any type to be executed on any one of a bank of processors by any one thread within a pool of threads, a transaction processing system such as that illustrated in **Figure 2** is able to re-distribute resources to threads within the pool of threads 202, and accordingly across a bank of processors 32, to serve specific peak performance demands. Further, scalability of the transaction processing system 30 is enhanced.

Figure 10 is a block diagram providing a conceptual illustration of the generation of a task 250, that is queued within the task queue 128, responsive to an event 146. Specifically, the event definitions 210, which are stored in the database server 124 shown in **Figure 5**, provide input to both the identification of the event 146 and into the workflow 215 through the event-workflow bindings 214, which are also stored in the database server 124. The workflow 215 may, for example, include the workflow definitions 208 and the workflow priority information 216 stored in the database server 124. The event 146 is then compared to an event rules set 217, which may include the event-workflow bindings 214, to identify a workflow for the given event, considering the event type and the event content. The workflow 215 is then shown to be instantiated as a set of tasks 250 that are propagated by the re-entrant task dispatchers 200 called by the various event subsystems to the task queue 128.

Accordingly, a method and apparatus for executing a transaction task within a transaction processing system employing a multiprocessor architecture have been described. Although the present invention has been described with reference to specific exemplary embodiments, it will be
5 evident that various modifications and changes may be made to these embodiments without departing from the broader spirit and scope of the invention. Accordingly, the specification and drawings are to be regarded in illustrative rather than a restrictive sense.

CLAIMS

What is claimed is:

- 1 1. A method of executing a transaction task within a transaction
2 processing system, the method including:
3
4 responsive to an event, identifying a workflow associated with the
5 event; and
6
7 distributing a task, that at least partially executes the workflow, to an
8 available thread within a pool of threads operating within a
9 multiprocessor system.
- 1 2. The method of claim 1 wherein the event comprises a transaction
2 event and the task comprises a transaction task responsive to a transaction
3 request associated with the transaction event.
- 1 3. The method of claim 2 wherein the transaction task comprises a
2 transaction routing task that routes the transaction request associated with
3 the transaction event to an agent of the transaction processing system.

1 4. The method of claim 2 within the transaction task comprises a
2 transaction information task to either store or retrieve information pertinent
3 to a transaction.

1 5. The method of claim 1 wherein the task has a real-time priority and is
2 distributed in accordance with the real-time priority to the available thread
3 within the pool of threads.

1 6. The method of claim 1 including identifying a processor affinity
2 attributed to the task, and assigning the available thread to a processor
3 within the multiprocessor system according to the processor affinity
4 attributed to the task.

1 7. The method of claim 1 including assigning the available thread to a
2 processor within the multiprocessor system according to a thread priority.

1 8. The method of claim 7 including assigning the thread priority to the
2 available thread based on a priority of the task distributed to the available
3 thread.

1 9. Apparatus for executing a transaction task within a transaction
2 processing system, the apparatus comprising:
3

4 a dispatcher to identify a workflow associated with an event; and
5
6 a thread within a pool of threads operating within a multiprocessor
7 system to execute a task that at least partially executes the workflow
8 associated with the event.

1 10. The apparatus of claim 9 wherein the dispatcher generates the task
2 that at least partially executes the workflow.

1 11. The apparatus of claim 10 including a task queue to which the task is
2 dispatch by the dispatcher, and from which the thread within the pool of
3 threads receives the task.

1 12. The apparatus of claim 11 including a scheduler that issues the task
2 from the task queue to the thread within the pool of threads.

1 13. The apparatus of claim 12 wherein the scheduler issues the task from
2 the task queue to the thread within the pool of threads based on a priority
3 associated with the task.

1 14. The apparatus of claim 13 wherein the scheduler issues the task from
2 the task queue according to a priority dynamically assigned to the task.

1 15. The apparatus of claim 13 wherein the scheduler issues the task from
2 the task queue according to a real-time priority assigned to the task.

1 16. The apparatus of claim 9 wherein the task comprises a transaction
2 routing task that routes a transaction request associated with the event to an
3 agent of the transaction processing system.

1 17. The apparatus of claim 9 within the task comprises a transaction
2 information task to either store or retrieve information pertinent to a
3 transaction.

1 18. The apparatus of claim 9 including a dispatcher to identify a
2 processor affinity attributed to the task, and to assign the thread to a
3 processor within the multiprocessor system according to the processor
4 affinity attributed to the task.

1 19. The apparatus of claim 9 including to assign the thread to a processor
2 within the multiprocessor system according to a thread priority.

1 20. The apparatus of claim 19 including assigning the thread priority to
2 the thread based on a priority of the task distributed to the thread.

1 21. A method of operating a transaction processing system employing a

2 multiprocessor architecture, the method including:
3
4 establishing a queue of tasks, the queue of tasks including tasks for
5 both system and transactional functions, and
6
7 servicing the queue of tasks utilizing a pool of threads executable
8 within a systematic multiprocessor environment.

1 22. The method of claim 21 wherein the tasks for the system functions
2 include any one of reporting, administration or maintenance tasks
3 performed within the transaction processing system.

1 23. The method of claim 21 wherein the tasks for the transactional
2 functions include any one of routing, transaction data storage or transaction
3 data retrieval tasks performed to facilitate a transaction within the
4 transaction processing system.

ABSTRACT OF THE DISCLOSURE

A method of executing a transaction task within a transaction processing system includes, responsive to an event, the steps of identifying a workflow associated with the event. A transaction task, that at least
5 partially executes the workflow, is distributed to an available thread within a pool threads operating within a multiprocessor system, that may be a Symmetrical Multiprocessor (SMP) system.

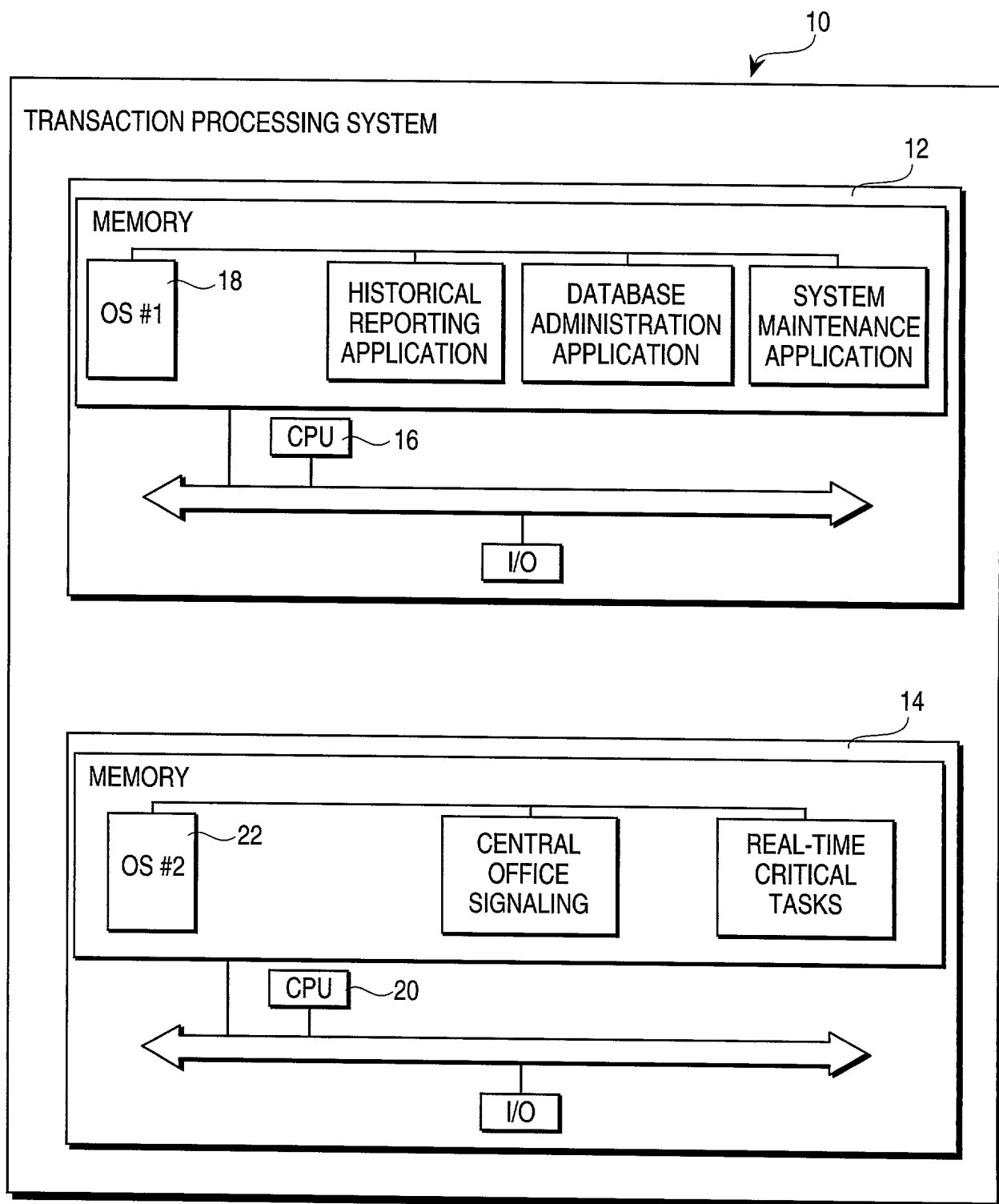


FIG. 1

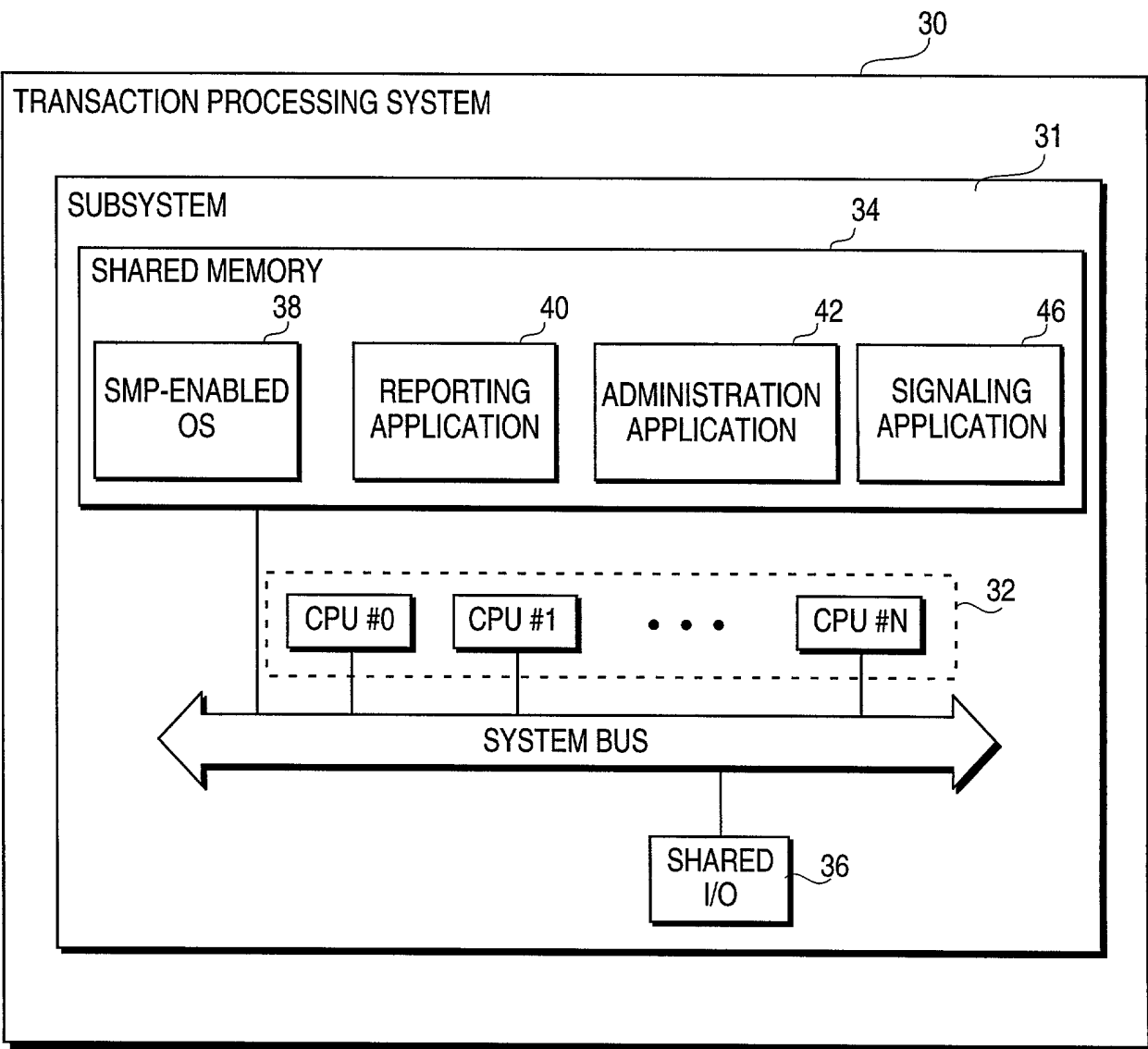


FIG. 2

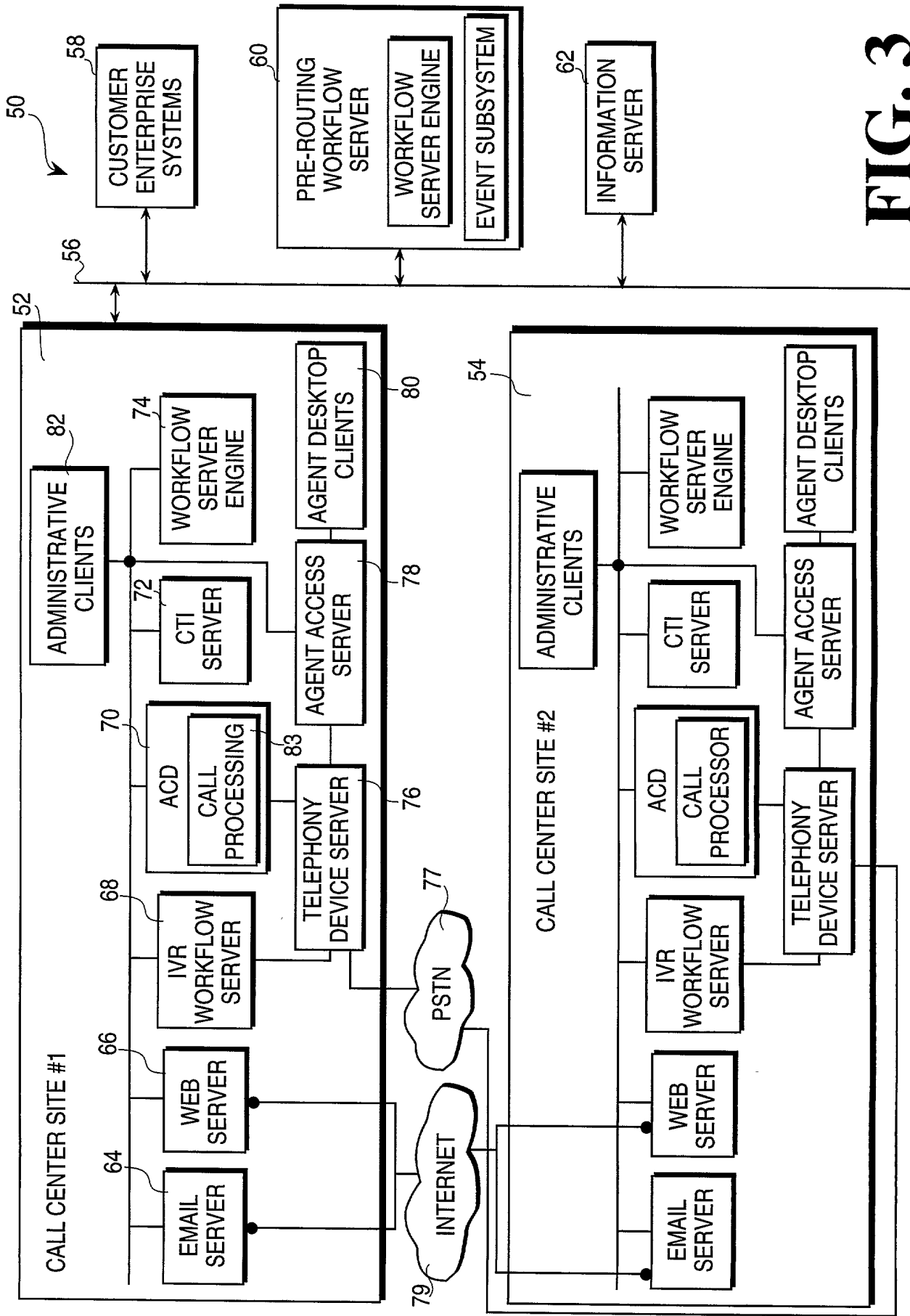


FIG. 3

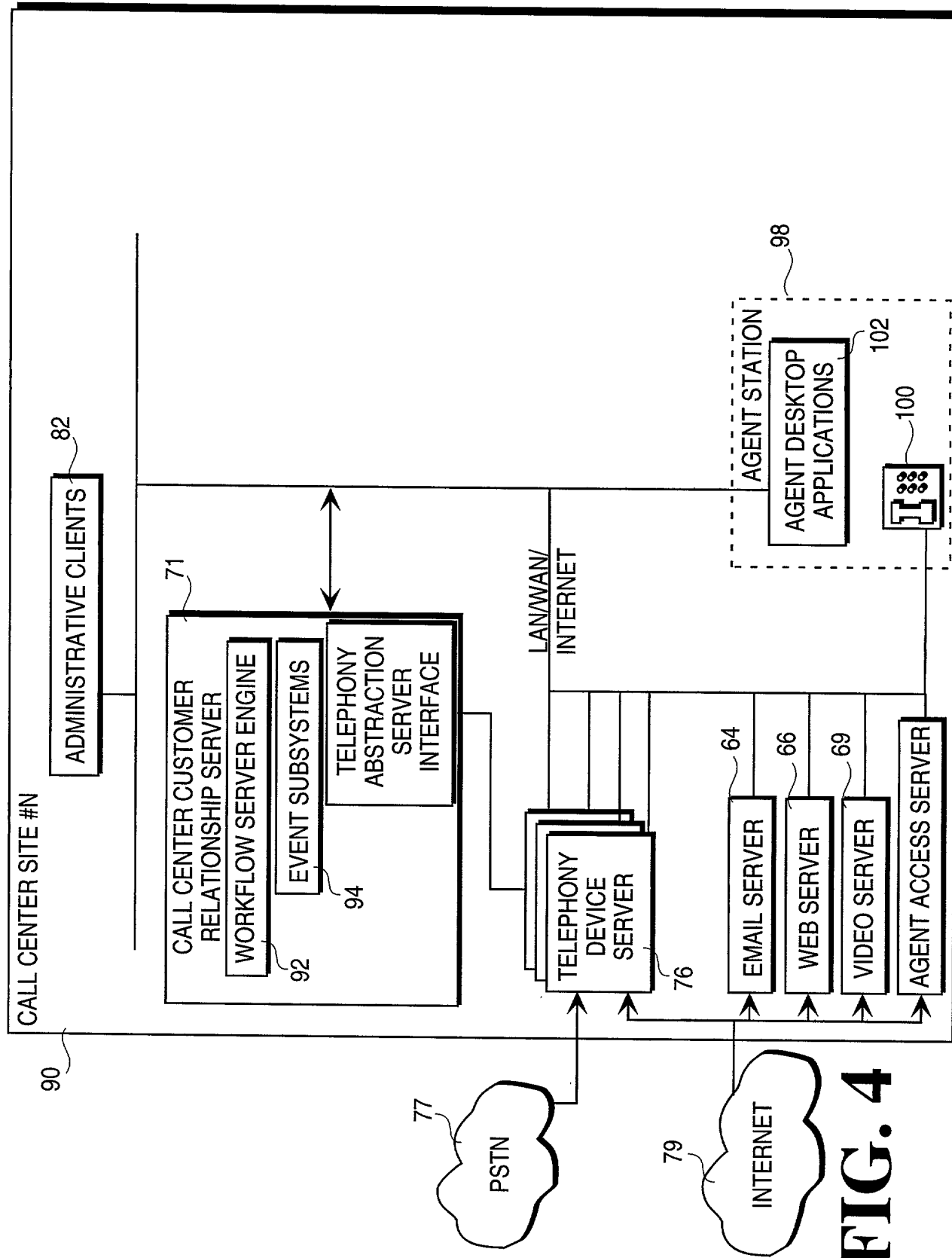


FIG. 4

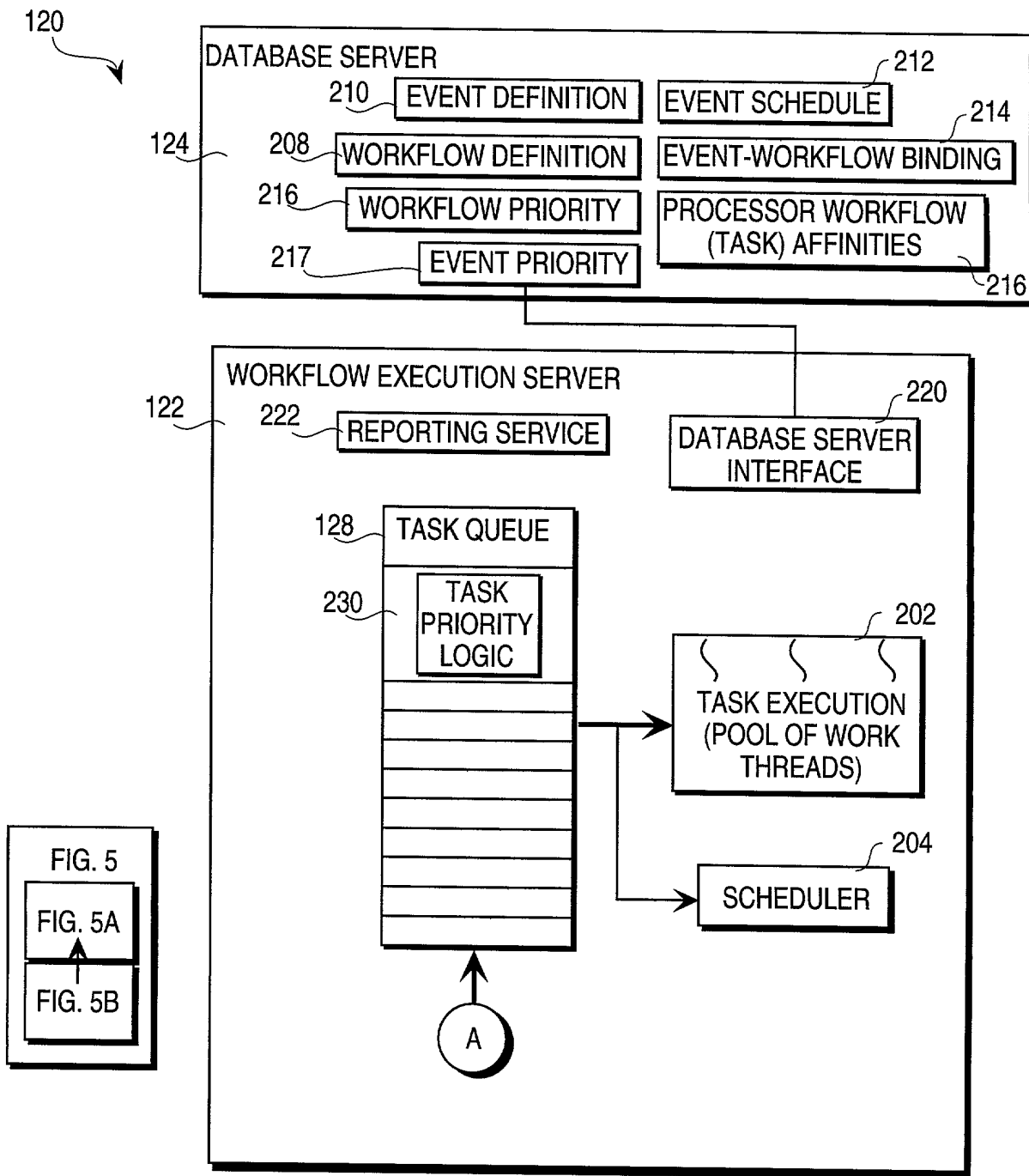


FIG. 5A

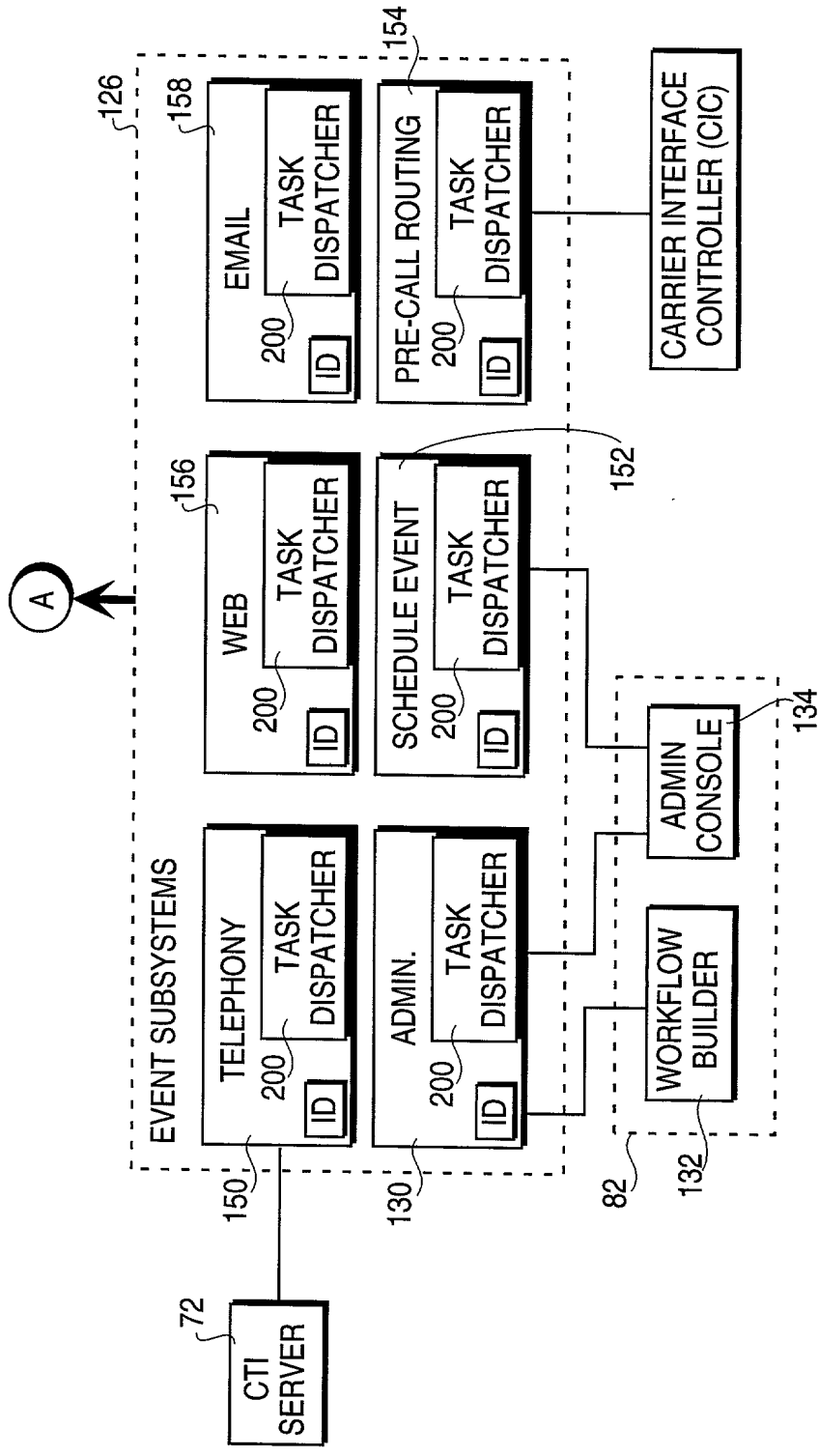


FIG. 5B

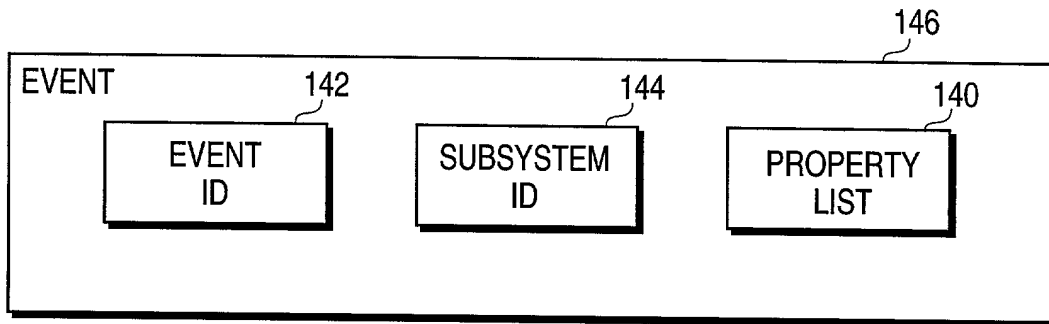


FIG. 6

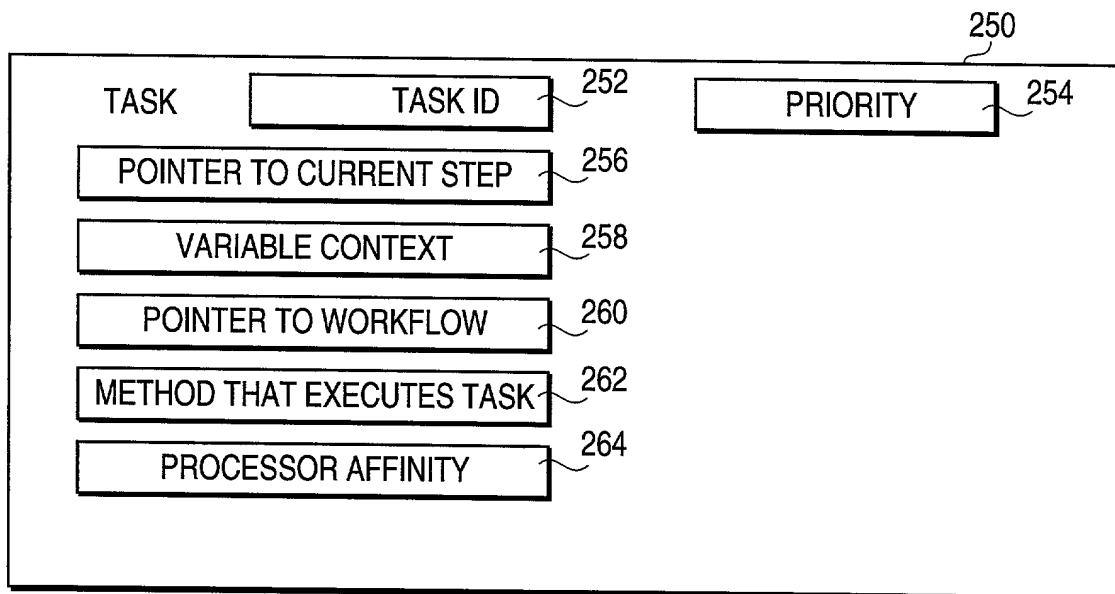


FIG. 7

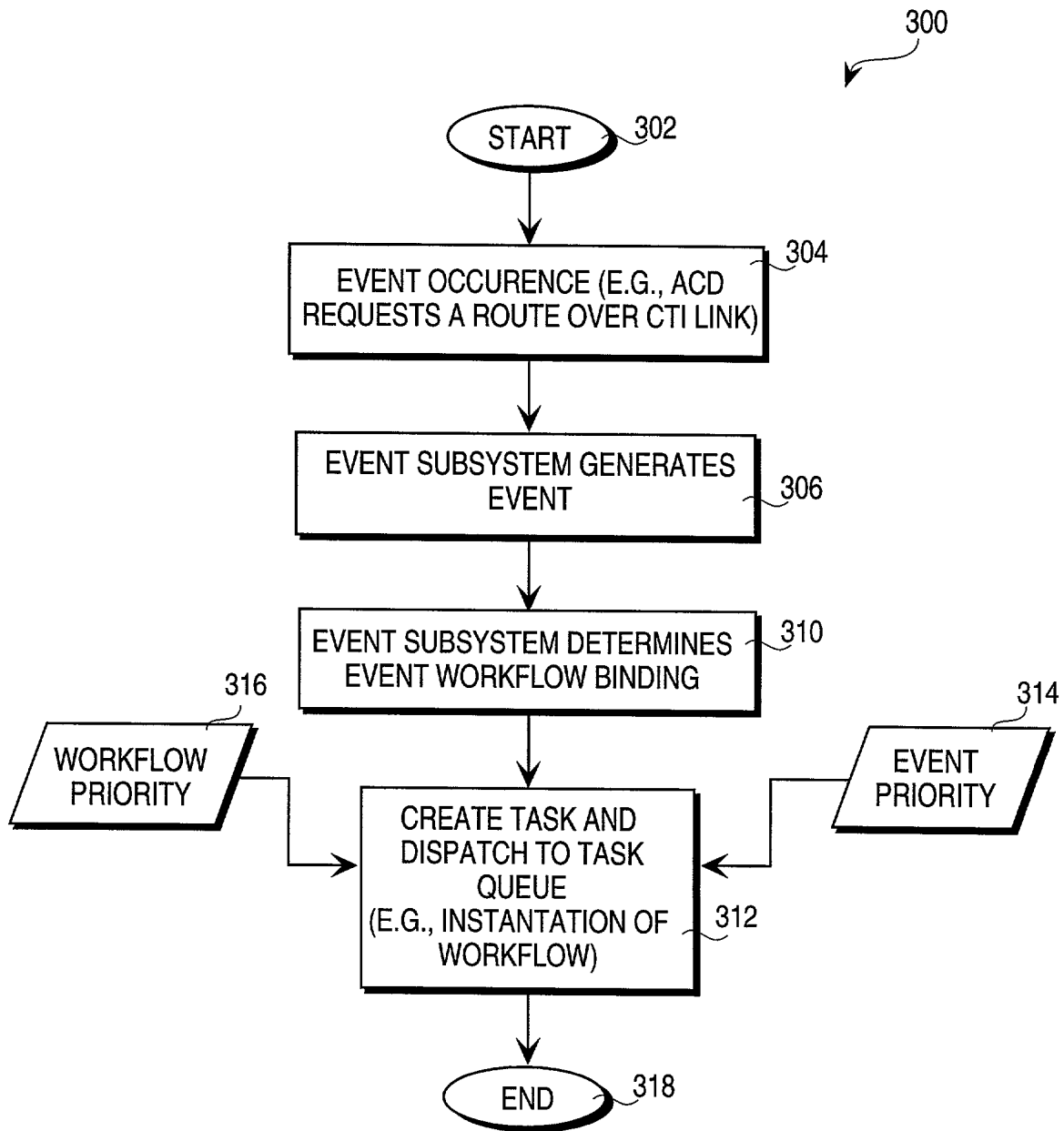


FIG. 8

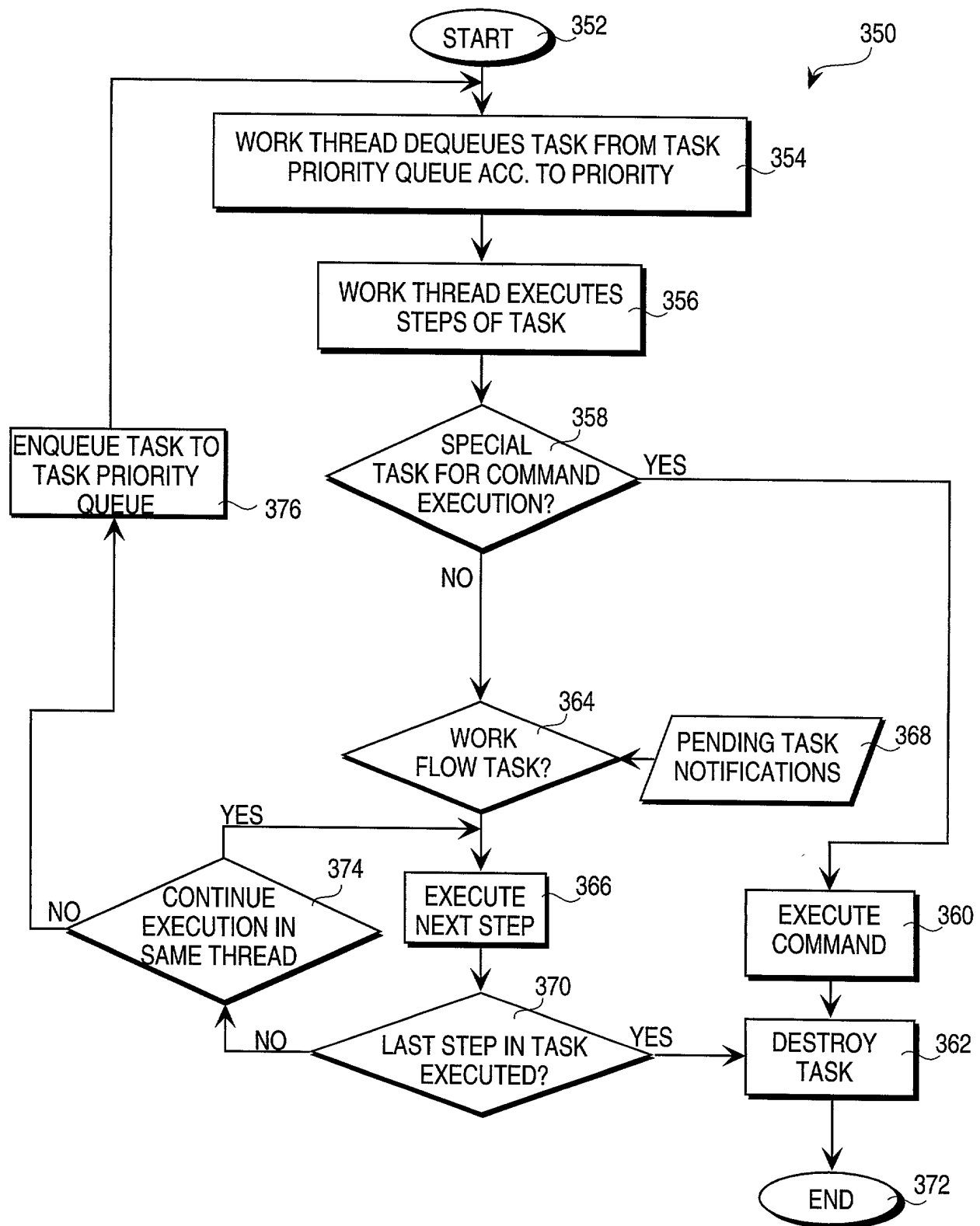


FIG. 9

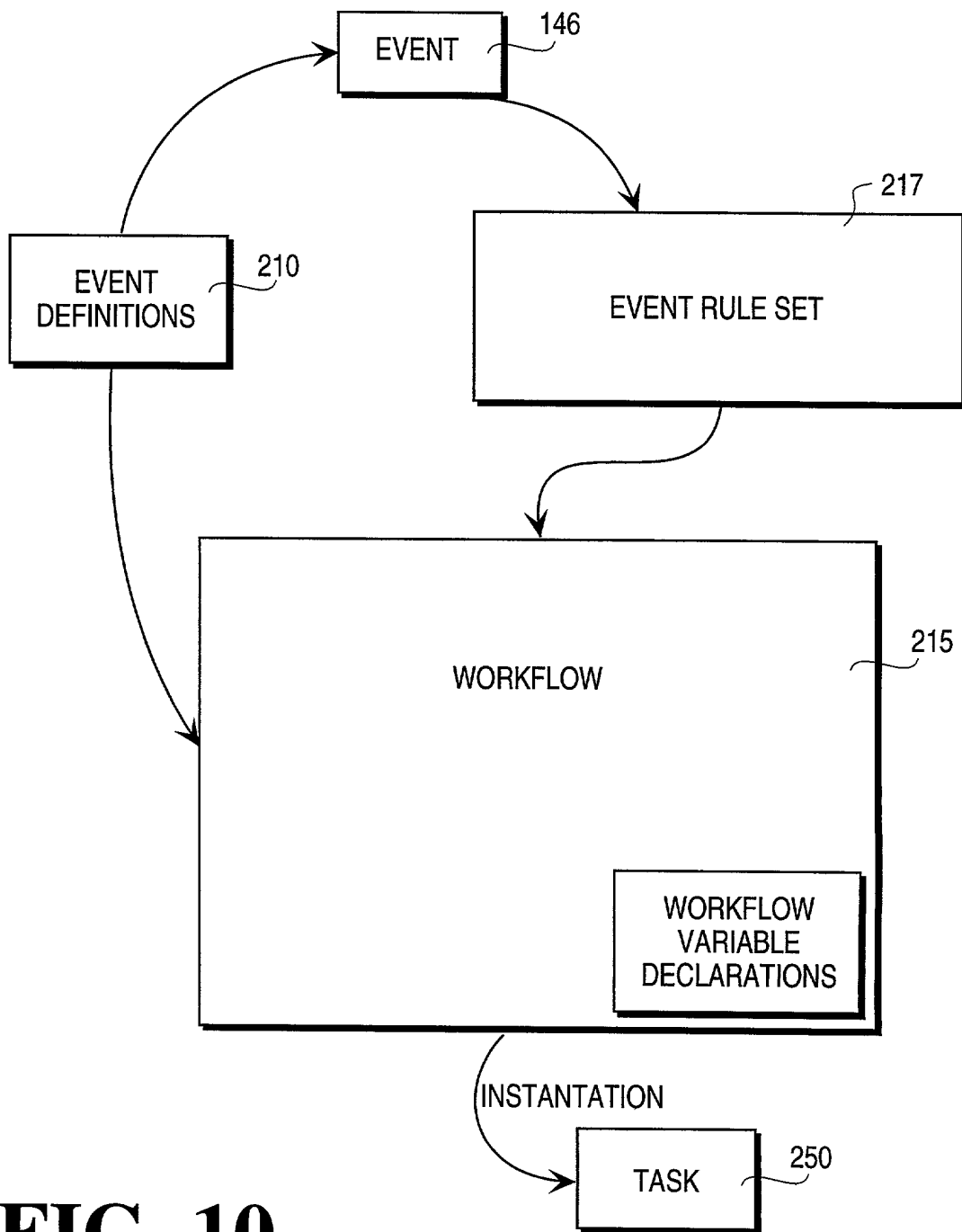


FIG. 10

As a below named inventor, I hereby declare that:

I believe I am the original, first, and sole inventor (if only one name is listed below) or an original, first, and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

the specification of which

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claim(s), as amended by any amendment referred to above.

I acknowledge the duty to disclose all information known to me to be material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119(a)-(d), of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

<u>Prior Foreign Application(s)</u>			<u>Priority Claimed</u>	
<u>(Number)</u>	<u>(Country)</u>	<u>(Day/Month/Year Filed)</u>	<u>Yes</u>	<u>No</u>
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No

I hereby claim the benefit under title 35, United States Code, Section 119(e) of any United States provisional application(s) listed below

(Application Number)	Filing Date
(Application Number)	Filing Date

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose all information known to me to be material to patentability as defined in Title 37, Code of Federal Regulations, Section 1.56 which became available between the filing date of the prior application and the national or PCT international filing date of this application:

(Application Number)	Filing Date	(Status -- patented, pending, abandoned)
(Application Number)	Filing Date	(Status -- patented, pending, abandoned)

I hereby appoint Farzad E. Amini, Reg. No. P42,261; Aloysius T. C. AuYeung, Reg. No. 35,432; Amy M. Armstrong, Reg. No. 42,265; William Thomas Babbitt, Reg. No. 39,591; Carol F. Barry, Reg. No. 41,600; Jordan Michael Becker, Reg. No. 39,602; Bradley J. Bereznak, Reg. No. 33,474; Michael A. Bernadicou, Reg. No. 35,934; Roger W. Blakely, Jr., Reg. No. 25,831; Gregory D. Caldwell, Reg. No. 39,926; Kent M. Chen, Reg. No. 39,630; Lawrence M. Cho, Reg. No. 39,942; Yong S. Choi, Reg. No. P43,324; Thomas M. Coester, Reg. No. 39,637; Roland B. Cortes, Reg. No. 39,152; Barbara Bokanov Courtney, Reg. No. 42,442; Michael Anthony DeSanctis, Reg. No. 39,957; Daniel M. De Vos, Reg. No. 37,813; Robert Andrew Diehl, Reg. No. 40,992; Tarek N. Fahmi, Reg. No. 41,402; James Y. Go, Reg. No. 40,621; Richard Leon Gregory, Jr., Reg. No. 42,607; Dinu Gruia, Reg. No. P42,996; David R. Halvorson, Reg. No. 33,395; Thomas A. Hassing, Reg. No. 36,159; Phuong-Quan Hoang, Reg. No. 41,839; Willmore F. Holbrow III, Reg. No. P41,845; George W. Hoover II, Reg. No. 32,992; Eric S. Hyman, Reg. No. 30,139; Dag H. Johansen, Reg. No. 36,172; William W. Kidd, Reg. No. 31,772; Michael J. Mallie, Reg. No. 36,591; Andre L. Marais, under 37 C.F.R. § 10.9(b); Paul A. Mendonsa, Reg. No. 42,879; Darren J. Milliken, Reg. No. 42,004; Thinh V. Nguyen, Reg. No. 42,034; Kimberley G. Nobles, Reg. No. 38,255; Michael A. Proksch, Reg. No. 43,021; Babak Redjaian, Reg. No. 42,096; James H. Salter, Reg. No. 35,668; William W. Schaal, Reg. No. 39,018; James C. Scheller, Reg. No. 31,195; Anand Sethuraman, Reg. No. P43,351; Charles E. Shemwell, Reg. No. 40,171; Maria McCormack Sobrino, Reg. No. 31,639; Stanley W. Sokoloff, Reg. No. 25,128; Allan T. Sponseller, Reg. No. 38,318; Judith A. Szepesi, Reg. No. 39,393; Vincent P. Tassinari, Reg. No. 42,179; Edwin H. Taylor, Reg. No. 25,129; George G. C. Tseng, Reg. No. 41,355; Lester J. Vincent, Reg. No. 31,460; John Patrick Ward, Reg. No. 40,216; Stephen Warhola, Reg. No. 43,237; Charles T. J. Weigell, Reg. No. 43,398; Ben J. Yorks, Reg. No. 33,609; and Norman Zafman, Reg. No. 26,250; my attorneys, and James A. Henry, Reg. No. 41,064; Daniel E. Ovanezian, Reg. No. 41,236; Glenn E. Von Tersch, Reg. No. 41,364; and Chad R. Walsh, Reg. No. 43,235; my patent agents, of BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP, with offices located at 12400 Wilshire Boulevard, 7th Floor, Los Angeles, California 90025, telephone (310) 207-3800, and James R. Thein, Reg. No. 31,710, my patent attorney; with full power of substitution and revocation, to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith.

Send correspondence to André L. Marais, BLAKELY, SOKOLOFF, TAYLOR &
(Name of Attorney or Agent)
ZAFMAN LLP, 12400 Wilshire Boulevard 7th Floor, Los Angeles, California 90025 and direct
telephone calls to André L. Marais, (408) 720-8598.
(Name of Attorney or Agent)

I hereby declare that all statements made herein of my own knowledge are true and that all
statements made on information and belief are believed to be true; and further that these
statements were made with the knowledge that willful false statements and the like so made are
punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States
Code and that such willful false statements may jeopardize the validity of the application or any
patent issued thereon.

Full Name of Sole/First Inventor Paul Evan Matz

Inventor's Signature _____ Date _____

Residence San Francisco, California Citizenship USA
(City, State) (Country)

Post Office Address 4054 20th St.
San Francisco, CA 94114

Full Name of Second/Joint Inventor Glen K. Okita

Inventor's Signature _____ Date _____

Residence San Jose, California Citizenship USA
(City, State) (Country)

Post Office Address 1327 Maria Way
San Jose, CA 95117-3618

Full Name of Third/Joint Inventor Gebran Chahrouri

Inventor's Signature _____ Date _____

Residence Menlo Park, California Citizenship USA
(City, State) (Country)

Post Office Address 11318 Marcussen Drive
Menlo Park, CA 94025

Full Name of Fourth/Joint Inventor Michael Butensky

Inventor's Signature _____ Date _____

Residence Los Altos, California Citizenship USA
(City, State) (Country)

Post Office Address 66 N. Avalon Drive
Los Altos, CA 94022

Title 37, Code of Federal Regulations, Section 1.56
Duty to Disclose Information Material to Patentability

(a) A patent by its very nature is affected with a public interest. The public interest is best served, and the most effective patent examination occurs when, at the time an application is being examined, the Office is aware of and evaluates the teachings of all information material to patentability. Each individual associated with the filing and prosecution of a patent application has a duty of candor and good faith in dealing with the Office, which includes a duty to disclose to the Office all information known to that individual to be material to patentability as defined in this section. The duty to disclose information exists with respect to each pending claim until the claim is cancelled or withdrawn from consideration, or the application becomes abandoned. Information material to the patentability of a claim that is cancelled or withdrawn from consideration need not be submitted if the information is not material to the patentability of any claim remaining under consideration in the application. There is no duty to submit information which is not material to the patentability of any existing claim. The duty to disclose all information known to be material to patentability is deemed to be satisfied if all information known to be material to patentability of any claim issued in a patent was cited by the Office or submitted to the Office in the manner prescribed by §§1.97(b)-(d) and 1.98. However, no patent will be granted on an application in connection with which fraud on the Office was practiced or attempted or the duty of disclosure was violated through bad faith or intentional misconduct. The Office encourages applicants to carefully examine:

(1) Prior art cited in search reports of a foreign patent office in a counterpart application, and

(2) The closest information over which individuals associated with the filing or prosecution of a patent application believe any pending claim patentably defines, to make sure that any material information contained therein is disclosed to the Office.

(b) Under this section, information is material to patentability when it is not cumulative to information already of record or being made of record in the application, and

(1) It establishes, by itself or in combination with other information, a prima facie case of unpatentability of a claim; or

(2) It refutes, or is inconsistent with, a position the applicant takes in:

(i) Opposing an argument of unpatentability relied on by the Office, or

(ii) Asserting an argument of patentability.

A prima facie case of unpatentability is established when the information compels a conclusion that a claim is unpatentable under the preponderance of evidence, burden-of-proof standard, giving each term in the claim its broadest reasonable construction consistent with the specification, and before any consideration is given to evidence which may be submitted in an attempt to establish a contrary conclusion of patentability.

(c) Individuals associated with the filing or prosecution of a patent application within the meaning of this section are:

(1) Each inventor named in the application;

(2) Each attorney or agent who prepares or prosecutes the application; and

(3) Every other person who is substantively involved in the preparation or prosecution of the application and who is associated with the inventor, with the assignee or with anyone to whom there is an obligation to assign the application.

(d) Individuals other than the attorney, agent or inventor may comply with this section by disclosing information to the attorney, agent, or inventor.

02950.P033

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:

Paul E. Matz, et al.

Examiner: Not yet assigned

Serial No.: New application

Art Unit: Not yet assigned

Filing Date: Herewith

For: METHODS AND APPARATUS FOR
EXECUTING A TRANSACTION
TASK WITHIN A TRANSACTION
PROCESSING SYSTEM EMPLOYING
SYMMETRIC MULTIPROCESSORS

Assistant Commissioner for Patents
Washington, D.C. 20231

APPOINTMENT OF ASSOCIATE ATTORNEY

Sir:

I hereby appoint André L. Marais as my associate attorney in the above-entitled application, to prosecute this application, to make alterations and amendments therein, and to transact all business in the Patent and Trademark Office connected therewith.

Please continue to address all future communications to Blakely, Sokoloff, Taylor & Zafman LLP, 12400 Wilshire Blvd., Seventh Floor, Los Angeles, CA 90025-1026.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN, LLP

Date: 5-26, 1999

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, CA 90025-1026
(408) 720-8598



James Y. Go
Registration No. 40,621